Cellular strategies and innocent interaction

Russ Harmer

Cellular strategies and innocent interaction

Russ Harmer
PPS, CNRS & Université de Paris 7

March 26, 2007

Abstract

An analysis of the process of interaction between innocent strategies is presented, leading to a new class of *cellular* strategies with an associated abstract machine, the CPAM, which radically simplifies the usual machines for innocent interaction. A *cellularization* process, mapping an innocent strategy to a cellular strategy, is then described which allows us to simulate innocent interaction by first cellularizing both strategies then running them in the CPAM.

1 Introduction

In game semantics, computation is modelled as a sequence of moves, a *dialogue*, between two protagonists: Player and Opponent. A particular program is then modelled as a *strategy* that tells Player—who represents the program—how to respond to all the moves that could be made by Opponent—who represents the context. A strategy can thus be thought of as a set of admissible dialogues where Player must respect the strategy but Opponent can play freely. We have a natural way of making two strategies interact: each plays the role of Opponent for the other. So, in an interaction, both Player and Opponent must respect a strategy.

In general, the choice of move played by a strategy at a given moment depends on the whole of the dialogue to date. However, the constraint of *innocence* [4, 6] imposed on the game semantics of PCF (and of μPCF) requires that a strategy's behaviour depends on only a *subsequence of the dialogue to date*. As a result, when two innocent strategies interact, a certain amount of book-keeping becomes necessary:

Each time one strategy makes a move, the other strategy [to play next] must have access to all, and only, the information to which it is entitled [in order to make that next move]. Unfortunately, the first strategy *may not have access to all this information*, so it seems that we need an intermediary, a *referee*, which repeatedly sends one strategy its next input, receives its response and then, based on that response, calculates the next input for the other strategy.

This vision of interaction makes interacting innocent strategies completely subservient to the referee. Indeed, it concentrates all the intelligence of the interaction in the referee, relegating the strategies to a role of merely reacting to whatever stimulus the referee decides to send them. Could we, at least to some extent, "eliminate the referee", *i.e.* delegate to the strategies some or all of the referee's responsibilities?

To answer this question, we describe two abstract machines. The first, the DPAM, implements innocent interaction as described above, using a referee; the second, the CPAM, implements interaction for a new class of *cellular* strategies and needs no referee. We then describe how to transform an innocent strategy σ into a cellular strategy $\sigma^{©}$ in such a way that the DPAM interaction between σ and τ is simulated by the CPAM interaction between $\sigma^{©}$ and $\tau^{©}$. We can, therefore, eliminate the referee—albeit at the cost of cellularizing.

Acknowledgements Thanks to Pierre Clairambault, Pierre-Louis Curien, Thierry Joly and Vincent Padovani for numerous discussions related to the subject of this paper.

2 The CCC of innocent strategies

In this section, we briefly present the definitions of HO-style game semantics that lead to the fundamental CCC of *arenas and innocent strategies*.

2.1 The objects

2.1.1 Strings with pointers

Let Σ be a countable set. A ***string-with-pointers over*** Σ is a string $s \in \Sigma^\star$ with pointers between the symbol-occurrences of s such that

- there is at most one pointer from any given symbol-occurrence of s
- if s_i [the ith symbol-occurrence of s] points to s_j, then $j < i$.

In words, each symbol-occurrence of s has either no associated pointer or exactly one pointer pointing to an earlier symbol-occurrence of the string. We could formalize the pointers on a string of length n as a strictly decreasing partial function on $\{1, \ldots, n\}$; but, for simplicity, we mention this as little as possible.

We write $|s|$ for the ***length*** of s and, if a is a symbol-occurrence in s, we write $s_{\leqslant a}$ (resp. $s_{<a}$) for the ***truncation*** of s at a including (resp. excluding) a. The ***prefix*** ordering on strings extends to strings-with-pointers s and t in the following way: $s \sqsubseteq t$ iff $s = t$ or, for some symbol-occurrence a in t, $s = t_{<a}$. The set of all strings-with-pointers over Σ is clearly a poset under this ordering with least element ε, the empty string(-with-pointers).

If $\Sigma' \subseteq \Sigma$ then we write $s \restriction_{\Sigma'}$ for the ***restriction*** of s to Σ' obtained by erasing those symbol-occurrences of s from $\Sigma - \Sigma'$ and manipulating the pointers as follows: if the pointer from $s_i \in \Sigma'$ points into the "forbidden zone" $\Sigma - \Sigma'$, we follow the chain of pointers leading back from s_i until we either "reemerge" by pointing to some $s_z \in \Sigma'$, in which case s_i points to s_z in $s \restriction_{\Sigma'}$, or we run out of pointers, in which case s_i has no pointer in $s \restriction_{\Sigma'}$. In terms of the decreasing partial function representation, we partition the indices of symbol-occurrences of s into those from Σ' and $\Sigma - \Sigma'$ and take the trace to eliminate $\Sigma - \Sigma'$.

2.1.2 Arenas

In game semantics, two protagonists known as Player (P) and Opponent (O) play the role of programs and their contexts respectively. Computation is modelled by a dialogue between O and P which takes place in an *arena*:

An ***arena*** A is a tuple $\langle M_A, \lambda_A, I_A, \vdash_A \rangle$ where

- M_A is a countable set of ***tokens***
- $\lambda_A : M_A \to \{\mathsf{O}, \mathsf{P}\}$ is a function, known as ***labelling***, that designates each $m \in M_A$ as either an Opponent or Player token. We write $\overline{\lambda}_A$ for the *inverted* labelling that swaps Opponent and Player.
- I_A is a subset of $\lambda_A^{-1}(\mathsf{O})$ known as the ***initial moves*** of A
- $\vdash_A$ is a binary relation on M_A, known as ***enabling***, such that, if $m \vdash_A n$, then $n \notin I_A$ and $\lambda_A(m) \neq \lambda_A(n)$.

This definition partitions the set of tokens M_A into two disjoint pieces, O_A and P_A, with causality relations between, but not within, the two pieces. We often use $\circ$ or O as generic Opponent tokens, $\bullet$ or P for Player.

For example, the arena **bool** has one initial move q and two Player tokens, $\mathtt{tt}$ and $\mathtt{ff}$, where $\mathsf{q} \vdash_{\mathbf{bool}} \mathtt{tt}$ and $\mathsf{q} \vdash_{\mathbf{bool}} \mathtt{ff}$. More generally, we can replace the set $\{\mathtt{tt}, \mathtt{ff}\}$ with any countable set X. We call this the ***flat arena over*** X and write $\bot$, **com** and **nat** for the flat arenas over the empty set, a singleton and the natural numbers respectively. We write **1** for the ***empty*** arena $\langle \varnothing, \varnothing, \varnothing \rangle$.

2.1.3 Legal plays

An arena provides an abstract setting for strings-with-pointers. Specifically, a ***legal play*** in A is a string-with-pointers s over alphabet M_A, which satisfies

- OP***-alternation***: $\lambda_A(s_i) \neq \lambda_A(s_{i+1})$, for $1 \leq i < |s|$
- ***justification***: if s_i points to s_j then $s_j \vdash_A s_i$, for $1 < i \leq |s|$
- if s_i has no pointer then $s_i \in I_A$, for $1 \leq i \leq |s|$.

Legal plays formalize the notion of a dialogue between Opponent and Player. By convention, Opponent always starts, with an initial move, and the protagonists take turns thereafter. The pointers specify causality: a token (other than an initial move) can only be played by one protagonist if the other has already played some other token that enables it. The justification condition forces all tokens (except initial moves) to have such a pointer. We call a token together with its pointer a ***move*** of the legal play. We write $\mathcal{L}_A$ (resp. $\mathcal{L}_A^{\mathsf{P}}$, $\mathcal{L}_A^{\mathsf{O}}$) for the set of all (resp. P-ending, O-ending) legal plays of A where, by convention, ε is considered a P-ending play.

For $s, t \in \mathcal{L}_A$, we write $s \sqsubseteq^{\mathsf{P}} t$ (resp. $s \sqsubseteq^{\mathsf{O}} t$) iff $s \sqsubseteq t$ and $s \in \mathcal{L}_A^{\mathsf{P}}$ (resp. $s \in \mathcal{L}_A^{\mathsf{O}}$); $s \wedge t$ for the ***longest common prefix*** of s and t; s^- for the ***immediate prefix*** of non-empty s, *i.e.* s minus its last move; and s^+ for the set of ***immediate extensions*** of s, *i.e.* $\{t \mid s = t^-\}$; $s^{\leftarrow}$ for the ***justifying prefix*** of s ending with a non-initial move, *i.e.* the truncation of s at the move justifying its last move; $s^{\rightarrow}$ for the set of ***justifying extensions*** of s, *i.e.* $\{t \mid s = t^{\leftarrow}\}$; and, if $m \in M_A$ is enabled by the last move $s_{|s|}$ of s, $s \cdot m$ denotes the ***extension of*** s ***by*** m where m points to the *last move* of s, *i.e.* $(s \cdot m)^- = s$. If we write sm, we mean s extended with the token m and must additionally specify its pointer.

Note that, although we have defined legal plays as strings-with-pointers, we could equivalently present them as trees-with-pointers where the tree structure represents the justification pointers (which need no longer explicitly appear) and sequencing (the notion of immediate prefix) is represented by *contingency pointers* (implicit in usual legal plays); we call these latter ***tree plays***. One can easily move between these two equivalent representations.

2.1.4 Constructors

The ***product*** $A \times B$ of arenas A and B is defined by:

- $M_{A \times B} = M_A + M_B$, the disjoint union
- $\lambda_{A \times B} = [\lambda_A, \lambda_B]$, the copairing
- $I_{A \times B} = I_A + I_B$
- $\mathsf{inl}(m) \vdash_{A \times B} \mathsf{inl}(n)$ iff $m \vdash_A n$;
  $\mathsf{inr}(m) \vdash_{A \times B} \mathsf{inr}(n)$ iff $m \vdash_B n$

Note how *all* the structure of $A \times B$ is inherited from its constituent arenas; we place the arenas side-by-side with no possibility of "interaction" between them. We write A^ω for the obvious product of countably many copies of A with itself.

In contrast, the other major construction on arenas, the ***arrow*** $A \Rightarrow B$, adds in some new structure. As for the product, the tokens come from the disjoint union of those of A and B, but with the labelling inverted in A to reflect its contravariant nature. As a result, the initial moves of A become P-moves and can no longer be initial in $A \Rightarrow B$; instead, the initial moves of B enable them:

- $M_{A \Rightarrow B} = M_A + M_B$
- $\lambda_{A \Rightarrow B} = [\overline{\lambda}_A, \lambda_B]$
- $I_{A \Rightarrow B} = I_B$
- $\mathsf{inl}(m) \vdash_{A \Rightarrow B} \mathsf{inl}(n)$ iff $m \vdash_A n$;
  $\mathsf{inr}(m) \vdash_{A \Rightarrow B} \mathsf{inr}(n)$ iff $m \vdash_B n$;
  $\mathsf{inr}(m) \vdash_{A \Rightarrow B} \mathsf{inl}(n)$ iff $m \in I_B$ and $n \in I_A$

The flat arenas defined above can be expressed purely in terms of $\bot$ and the above constructors: $\mathbf{com} \cong \bot \Rightarrow \bot$; $\mathbf{bool} \cong (\bot \times \bot) \Rightarrow \bot$; and $\mathbf{nat} \cong \bot^{\omega} \Rightarrow \bot$.

2.2 The arrows

2.2.1 Strategies

In game semantics, arenas correspond to types; the set of legal plays of an arena describes *all* possible dialogues between programs (P) and contexts (O) of the type in question. A *particular* program is modelled by a subset of legal plays, a *strategy*, that specifies its (deterministic) response to every possible move of its (as yet unknown) context. More precisely, we associate to a program a set of P-ending legal plays with intended interpretation that if it contains *sab*, then after s, P continues by *playing b if* O *plays a*. By convention, all programs contain the empty play ε as starting point':

Formally, a ***strategy*** σ for an arena A, written $\sigma : A$, is a non-empty set of P-ending legal plays of A which satisfies

- P***-prefix-closure***: if $s \in \sigma$ and $s' \sqsubseteq^{\mathsf{P}} s$ then $s' \in \sigma$
- P***-determinism***: if $s \in \sigma$ and $t \in \sigma$ then $s \wedge t \in \sigma$.

The second condition amounts to asking for $s \wedge t$ to end with a P-move—so only O can branch nondeterministically. We write $\mathsf{dom}(\sigma)$ for the ***domain*** of σ defined to be $\bigcup_{s \in \sigma} s^+$, all the O-ending plays of A accessible to σ.

2.2.2 Legal interactions

In order to define the composition of two strategies $\sigma : A \Rightarrow B$ and $\tau : B \Rightarrow C$, we introduce the notion of a *legal interaction* of A, B and C:

Given arenas A, B and C, a ***legal interaction of*** A, B ***and*** C is a string-with-pointers u over $M_A + M_B + M_C$ such that $u \upharpoonright_{A,B} \in \mathcal{L}_{A \Rightarrow B}$, $u \upharpoonright_{B,C} \in \mathcal{L}_{B \Rightarrow C}$ and $u \upharpoonright_{A,C}$ satisfies OP-alternation. (It easily follows that $u \upharpoonright_{A,C} \in \mathcal{L}_{A \Rightarrow C}$.) We write $\mathcal{I}(A, B, C)$ for the set of all legal interactions of A, B and C. If A is the empty arena $\mathbf{1}$ and C a flat arena, we refer to $u \in \mathcal{I}(A, B, C)$ as a ***program interaction***.

We define the ***parallel composition*** of $\sigma : A \Rightarrow B$ and $\tau : B \Rightarrow C$ as

$$\sigma \mid \tau = \{u \in \mathcal{I}(A, B, C) \mid u \upharpoonright_{A,B} \in \sigma \wedge u \upharpoonright_{B,C} \in \tau\}$$

from which we hide the interaction in B to define their ***composition***:

$$\sigma ; \tau = \{u \upharpoonright_{A,C} \mid u \in \sigma \mid \tau\}.$$

This can be shown to be well-defined and associative and, for arena A, the ***copycat*** strategy $\mathsf{id}_A = \{s \in \mathcal{L}^{\mathsf{P}}_{A_\ell \Rightarrow A_r} \mid \forall s' \sqsubseteq^{\mathsf{P}} s.\ s' \upharpoonright_{A_\ell} = s' \upharpoonright_{A_r}\}$ for $A \Rightarrow A$ satisfies $\mathsf{id}_A ; \sigma = \sigma = \sigma ; \mathsf{id}_B$. So we have a category $\mathbf{G}$ with arenas as objects and strategies for $A \Rightarrow B$ as arrows from A to B.

We also write $\sigma \downarrow \tau$ for

$$\{u \in \mathcal{I}(A, B, C) \mid u \upharpoonright_{A,B} \in \sigma \cup \mathsf{dom}(\sigma) \ \wedge \ u \upharpoonright_{B,C} \in \tau \cup \mathsf{dom}(\tau)\}.$$

2.2.3 Monoidal structure

The product of two arenas lifts to a bifunctor on $\mathbf{G}$. Evident copycat strategies for $((A \times B) \times C) \Rightarrow (A \times (B \times C))$, $(A \times B) \Rightarrow (B \times A)$, $(\mathbf{1} \times A) \Rightarrow A$ and $(A \times \mathbf{1}) \Rightarrow A$ equip $\mathbf{G}$ with a symmetric monoidal structure.

A further evident copycat for $((A \times B) \Rightarrow C) \Rightarrow (A \Rightarrow (B \Rightarrow C))$ yields the *currying* isomorphism which we write as $\Lambda(-)$. Uncurrying (for all A and B) $\mathsf{id}_{A \Rightarrow B}$ yields $\epsilon_{A,B} : (A \Rightarrow B) \times A \Rightarrow B$, the *evaluation* maps, which satisfy, for any $\sigma : A \times B \Rightarrow C$, that $\sigma = (\Lambda(\sigma) \times \mathsf{id}_B) ; \epsilon_{B,C}$. This establishes that $\mathbf{G}$ is an SMCC (symmetric monoidal closed category).

2.3 The CCC

In this section, we introduce a subclass of strategies, the *innocent* strategies, whose behaviour depends only on a partial history of the play-to-date, the so-called *Player view* (or just P-view).

2.3.1 The Player view

The P-***view*** of a non-empty legal play $s \in \mathcal{L}_A$, noted $\ulcorner s \urcorner$, is defined in two stages. First we extract a subsequence of s with pointers defined only on (non-initial) O-moves:

- $\ulcorner s \urcorner = s_{|s|}$, if $s_{|s|}$ [the last move of s] is an initial move;
- $\ulcorner s \urcorner = \ulcorner s^{\leftarrow} \urcorner \cdot s_{|s|}$, if $s_{|s|}$ is a non-initial O-move;
- $\ulcorner s \urcorner = \ulcorner s^{-} \urcorner \ s_{|s|}$, if $s_{|s|}$ is a P-move.

In words, we trace back from the end of s, following pointers from O-moves, excising all moves under such pointers, and "stepping over" P-moves, until we reach an initial move, whereupon we stop. In general, a P-move m in s can *lose* its pointer: if its justifier n lies strictly underneath an O-to-P pointer of $\ulcorner s_{<m} \urcorner$ then n does not occur in $\ulcorner s_{<m} \urcorner$.

The second stage of the definition thus specifies that, if the justifier of a P-move in $\ulcorner s \urcorner$ gets excised in this way, it has *no justifier* in the P-view (and so $\ulcorner s \urcorner \notin \mathcal{L}_A$); otherwise it keeps the same justifier as in s.

A legal play $s \in \mathcal{L}_A$ satisfies P-***visibility*** iff $\ulcorner s \urcorner \in \mathcal{L}_A$. In words, no P-move of $\ulcorner s \urcorner$ loses its pointer. (Note that, for t some proper prefix of s, this doesn't prevent a P-move of $\ulcorner t \urcorner$ losing its pointer.)

We lift the definition of P-visibility to strategies in the obvious way: σ satisfies P-***visibility*** (or just 'is P-vis') iff all $s \in \sigma$ do. So, for s in P-vis σ as opposed to arbitrary P-vis s, all $t \sqsubseteq^{\mathsf{P}} s$ do in fact satisfy P-visibility, so $\ulcorner t \urcorner \in \mathcal{L}_A$ for all the P-prefixes t of s.

The notion of P-view generalizes naturally to legal interactions, from which we can easily show that P-vis strategies are closed by composition. The SMCC structure described above restricts without incident to the resulting subcategory of P-vis strategies.

2.3.2 Innocent strategies

Intuitively, the response of an innocent strategy σ to some $s \in \mathsf{dom}(\sigma)$ depends only on $\ulcorner s \urcorner$. We need the following simple fact about P-vis plays:

If $s \in \mathcal{L}_A^{\mathsf{P}}$ is P-vis, $t \in \mathcal{L}_A$ and $\ulcorner s^- \urcorner = \ulcorner t \urcorner$ then there exists a unique $t' \in t^+$ such that $\ulcorner s \urcorner = \ulcorner t' \urcorner$. We denote this t' by $\mathsf{match}_{\mathsf{P}}(s, t)$. The function $\mathsf{match}_{\mathsf{P}}$ allows us to formalize the idea that a strategy *responds to* s^- *and* t *in the same way, i.e.* by playing the same token, pointing to the same move of their *common* current P-view $\ulcorner s^- \urcorner$.

A P-vis strategy σ for A is ***innocent*** iff

$$s \in \sigma \ \wedge \ t \in \mathsf{dom}(\sigma) \ \wedge \ \ulcorner s^- \urcorner = \ulcorner t \urcorner \implies \mathsf{match}_{\mathsf{P}}(s, t) \in \sigma.$$

An innocent strategy is thus completely determined by its P-***view function*** $\ulcorner \sigma \urcorner = \{\ulcorner s \urcorner \mid s \in \sigma\}$: all legal plays of σ arise as *interleavings of entries of* $\ulcorner \sigma \urcorner$.

This observation leads to two equivalent characterizations of innocence as

$$\forall s \in \mathcal{L}_A.\ (s \in \sigma \iff \ulcorner s \urcorner \in \sigma)$$

or as

$$s \in \mathcal{L}_A \wedge s^- \in \mathsf{dom}(\sigma) \wedge \ulcorner s \urcorner \in \ulcorner \sigma \urcorner \ \Rightarrow \ s \in \sigma.$$

It can be shown, by extending $\mathsf{match}_{\mathsf{P}}$ to P-vis legal interactions, that innocent strategies are closed under composition. Since copycat strategies always satisfy innocence, we thus have a category $\mathbf{I}$ of arenas and innocent strategies. The monoidal structure of $\mathbf{G}$ restricts to a genuine product on $\mathbf{I}$, so $\mathbf{I}$ is a CCC.

2.3.3 The O-view

The O-***view*** of $s \in \mathcal{L}_A$, noted $\lfloor s \rfloor$, is defined dually to the P-view: follow pointers from P-moves and "step over" O-moves (so O-moves can lose pointers):

- $\lfloor \varepsilon \rfloor = \varepsilon$;
- $\lfloor s \rfloor = \lfloor s^{\leftarrow} \rfloor \cdot s_{|s|}$, if $s_{|s|}$ is a P-move;
- $\lfloor s \rfloor = \lfloor s^{-} \rfloor \; s_{|s|}$, if $s_{|s|}$ is an O-move.

In contrast to a P-view which always has a unique initial move, an O-view may contain many initial moves. A play $s \in \mathcal{L}_A$ satisfies O-***visibility*** (or just 'is O-vis') iff $\lfloor s \rfloor \in \mathcal{L}_A$, *i.e.* we lose no O-pointers in $\lfloor s \rfloor$. A strategy satisfies the ***visibility condition*** iff all of its plays satisfy P- and O-visibility.

When two P-vis strategies, $\sigma : A \Rightarrow B$ and $\tau : B \Rightarrow C$, interact, all moves played in B by τ (resp. σ) satisfy O-visibility in $A \Rightarrow B$ (resp. $B \Rightarrow C$). In other words, when we compose P-vis strategies, we lose no information if we restrict our attention to plays satisfying O-visibility. More precisely, if we write $\mathcal{O}(\sigma)$ for the set of all plays of σ that satisfy O-visibility, then

$$\mathcal{O}(\sigma \,;\, \tau) = \mathcal{O}(\mathcal{O}(\sigma) \,;\, \mathcal{O}(\tau)).$$

So O-vis plays of $\sigma \,;\, \tau$ come from interactions between O-vis plays of σ and τ. Henceforth, we assume (WLOG) that all legal plays satisfy O- and P-visibility.

2.3.4 The short O-view

The ***short*** O-***view*** $\llcorner s \lrcorner$ of non-empty s is a suffix of the O-view $\lfloor s \rfloor$ obtained by weakening the base case:

- $\llcorner s \lrcorner = s_{|s|}$, if $s_{|s|}$ points to an initial move;
- $\llcorner s \lrcorner = \llcorner s^{\leftarrow} \lrcorner \cdot s_{|s|}$, if $s_{|s|}$ is a P-move pointing to a non-initial move;
- $\llcorner s \lrcorner = \llcorner s^{-} \lrcorner \; s_{|s|}$, if $s_{|s|}$ is an O-move.

The short O-view $\llcorner s \lrcorner$ contains *no* initial moves—it stops "just short" of the first initial move m encountered by the O-view—and so cannot be a legal play. However, $m \llcorner s \lrcorner$ *is* a legal play (where the first move of $\llcorner s \lrcorner$ points to m). If $t \in \mathcal{L}_A^{\mathsf{P}}$, $s_{|s|}$ points in $\llcorner s \lrcorner$ and $\llcorner s^{-} \lrcorner = \llcorner t \lrcorner$ then we write $\mathsf{match}_{\mathsf{O}}(s, t)$ for the unique $t' \in t^{+}$ such that $\llcorner s \lrcorner = \llcorner t' \lrcorner$.

Consider a program interaction u between $\sigma : \mathbf{1} \Rightarrow A$ and $\tau : A \Rightarrow B$ with unique initial move i_u. By definition, $u \in \tau \cup \mathsf{dom}(\tau)$ and $u \upharpoonright_A \in \sigma \cup \mathsf{dom}(\sigma)$. For all $v \sqsubseteq u$ such that $v \in \tau$, the *next input* P-*view* for σ is $\llcorner v \lrcorner$; conversely, for all $v \sqsubseteq u$ such that $v \upharpoonright_A \in \sigma$, the next input P-view for τ is $i_u \lfloor v \upharpoonright_A \rfloor$. We refer to $\llcorner v \lrcorner$ (resp. $i_u \lfloor v \upharpoonright_A \rfloor$) as the ***output*** O-***view*** of τ (resp. σ) at v. So, σ needs the whole of the current O-view to determine its output O-view whereas τ needs only the current short O-view. This, in some sense, reflects the inherent asymmetry of interaction between a function and its argument(s).

2.4 Innocent interaction

2.4.1 Composing P-view functions

In general, the response of a strategy σ to some $s \in \mathsf{dom}(\sigma)$ depends on the whole of s. However, for innocent σ, it depends only on $\ulcorner s \urcorner$. This means that we have two different ways of composing innocent strategies: we either use the generic definition (but have to show that innocence is preserved by it—which it is, of course) or we can rephrase everything in terms of P-view functions.

The first approach seems better adapted to *defining* the category of innocent strategies; but, when we want to *calculate* the interaction between two innocent strategies (in an abstract machine or simply with pen and paper), we generally use the second approach: at each step, we read off the current P-view and then apply the P-view function (of the appropriate strategy) to it. However, a rather subtle problem arises with this second approach:

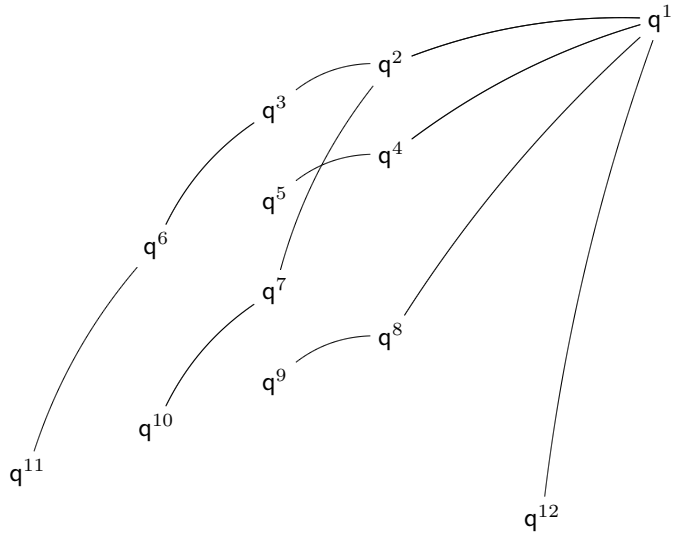
A P-view function receives an input P-view and adds a move to the end of it; the result is still a P-view, the ***output*** P***-view***. However, the *output* O*-view* cannot, in general, be deduced from this. To see why, consider the simply-typed (with unique base type o interpreted by the arena $\bot$) λ-terms

$$\mathcal{KK} \;=\; P : ((o \to o) \to o) \to o \;\vdash\; \lambda z^o(P)\lambda f(P)\lambda g(f)z : o \to o$$

$$\mathcal{K} \;=\; \;\vdash\; \lambda F(F)\lambda x(F)\lambda y(x) : ((o \to o) \to o) \to o$$

and the interaction (corresponding to $\mathcal{KK}[\mathcal{K}/P]$) between the respective innocent strategies [the superscripts serve only to identify moves]:

$$(((\bot \;\Rightarrow\; \bot) \;\Rightarrow\; \bot) \;\Rightarrow\; \bot) \quad\Rightarrow\quad (\bot \;\Rightarrow\; \bot)$$



After the third and the fifth moves, the output P-views of $\mathcal{K}$ are the same; yet, in the second case, the second and third moves must be inserted to get the correct output O-view. Similarly, after the sixth and the tenth moves, the output P-views of $\mathcal{K}\mathcal{K}$ are the same but, in the second case, the output P-view lacks $\mathsf{q}^3 \frown \mathsf{q}^6$ which must therefore be inserted to get the correct output O-view.

Of course, during a pen and paper calculation, *we* have no trouble "filling in" the missing chunks of output O-view because we write down the whole interaction, move-by-move, and can simply read off the output O-view after each move. But, when two P-view functions interact, certain information must be recorded so that the correct output O-views *can* be deduced from the output P-views (plus that information). In the next subsection, we briefly present the DPAM, a machine doing exactly this job.

2.4.2 The desequentialized PAM

Given two P-view functions, the DPAM makes them interact via a shared data structure, the *referee*. This data structure—a collection of nodes equipped with two distinct tree structures—is built as interaction proceeds. The two tree structures correspond to the justification pointers (of ordinary legal plays) and the contingency pointers (of tree plays).

In outline, the DPAM interrogates the two P-view functions alternately in order to build the referee data structure. Each time a P-view function makes a move, it adds a new node to the referee with two edges: the c-ptr and the j-ptr. It then offers this new node to the other P-view function.

When a P-view function is offered a node, it reads off the current P-view by traversing the referee: from its nodes, it follows the c-ptrs and from the other's, it follows the j-ptrs. We call this the ***access protocol***: a P-view function has read and write access to c-ptrs from its nodes, only write access to its j-ptrs and only read access to those of the other (and no access at all to the other's c-ptrs).

More formally, given P-view functions $\ulcorner\sigma\urcorner : A$ and $\ulcorner\tau\urcorner : A \Rightarrow B$ with B a flat arena, we build their ***referee*** recursively:

1. when a P-view function is offered a node, it reads off the current P-view s;
2. if the P-view function contains some t such that $s = t^-$ then it creates a new node n of the referee that encodes the last *token* of t, adds a *c-ptr* from n which points to the offered node, and a *j-ptr* for n, which points to that node of the referee encoding the justifier of n (in t);
3. it then offers n to the other view function.

This process stops when/if τ plays a move in B.

Note that, although the referee contains the whole interaction, neither P-view function can actually see this: the access protocol forbids any use of the other P-view function's c-ptrs. So, from the point of view of the P-view functions, the interaction really does appear to be desequentialized.

This clearly correctly implements innocent interaction:

Theorem 2.4.1 *Given innocent $\sigma : A$ and $\tau : A \Rightarrow B$ (flat arena B), the referee constructed by the DPAM for $\ulcorner\sigma\urcorner$ and $\ulcorner\tau\urcorner$ is the tree play equivalent of the increasing sequence of program interactions $u \in \sigma \downarrow \tau$.*

This machine unifies (and in some cases generalizes) many abstract machines in the literature [1–3, 5].

3 View-innocent and cellular strategies

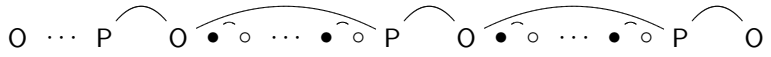
3.1 Views revisited

3.1.1 The OP-view

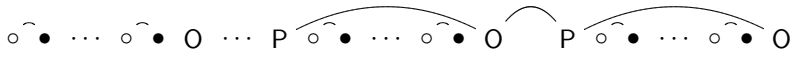
If s is a non-empty play that satisfies P- and O-visibility, we have $\ulcorner\lfloor s\rfloor\urcorner = \lfloor\ulcorner s\urcorner\rfloor$. We call $\lfloor\ulcorner s\urcorner\rfloor$ the OP-***view*** of s, which we write as $\lceil s\rceil$, and note that it simply consists of the chain of moves determined by following back pointers from the last move of s:

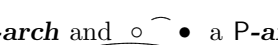
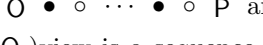
- $\lceil s\rceil = s_{|s|}$, if $s_{|s|}$ is an initial move;
- $\lceil s\rceil = \lceil s^{\leftarrow}\rceil \cdot s_{|s|}$, otherwise.

The OP-view appears in [6] as the "history of justification" and in [7] as the "bi-view". It can be seen as the *intersection* of the O- and the P-view. Inversely, we can see the P-view as an *annotation* where, underneath each P-to-O pointer of the OP-view, we have a sequence of O-to-P ($\bullet\frown\circ$) pointers [pointers from $\bullet$ moves elided for readability]:



and the O-view as the annotation of O-to-P pointers with sequences of P-to-O ($\circ\frown\bullet$) pointers [ditto for $\circ$ moves]:



We call $\bullet\frown\circ$ an O-***arch*** and $\circ\frown\bullet$ a P-***arch***;  an ***annotated*** O-***arch*** and  an ***annotated*** P-***arch***.

Note that a P- (resp. O-)view is a sequence of annotated P- (resp. O-)arches, linked by Opponent's (resp. Player's) justification pointers.

3.1.2 The view

In the light of the above remarks, it seems natural to superimpose the P-view and the O-view of a play, *i.e.* to annotate *all* arches of its OP-view. This constitutes an alternating sequence of annotated P- and O-arches whose last and first moves overlap:

We call this the *view* of s. To define it formally, we need the following facts:

- if $s \in \mathcal{L}_A^{\mathsf{P}}$, $t \in \mathcal{L}_A$ and $\ulcorner s^{\leftarrow} \urcorner = \ulcorner t \urcorner$ then there exists a unique $t' \in t^{\rightarrow}$ such that $\ulcorner s \urcorner = \ulcorner t' \urcorner$ and, for all $t'' \in \mathcal{L}_A^{\mathsf{P}}$ such that $t \sqsubseteq t'' \sqsubseteq t'$, $\ulcorner t'' \urcorner \sqsubseteq \ulcorner s \urcorner$.
- if $s \in \mathcal{L}_A^{\mathsf{O}}$, $t \in \mathcal{L}_A$ and $\lfloor s^{\leftarrow} \rfloor = \lfloor t \rfloor$ then there exists a unique $t' \in t^{\rightarrow}$ such that $\lfloor s \rfloor = \lfloor t' \rfloor$ and, for all $t'' \in \mathcal{L}_A^{\mathsf{O}}$ such that $t \sqsubseteq t'' \sqsubseteq t'$, $\lfloor t'' \rfloor \sqsubseteq \lfloor s \rfloor$.

We write $\mathsf{match}^{\star}(s,t)$ for t' (in both cases). In words, $\mathsf{match}^{\star}(s,t)$ extends t with the last annotated P-arch of $\ulcorner s \urcorner$ if s ends with a P-move; and with the last O-arch of $\lfloor s \rfloor$ if s ends with an O-move. The ***view*** of s, written $[s]$, is then defined by:

- $[s] = \lfloor s \rfloor$, if the last move of s is initial;
- $[s] = \mathsf{match}^{\star}(s, [s^{\leftarrow}])$, otherwise

Clearly, the view contains both the P-view and the O-view of s as subsequences.

A strategy is ***view-innocent*** iff

$$s \in \sigma \wedge t \in \mathsf{dom}(\sigma) \wedge [s^{-}] = [t] \implies \mathsf{match}_{\mathsf{P}}(s,t) \in \sigma.$$

View-innocent strategies are closed under composition, forming a subcategory $\mathbf{V}$ of $\mathbf{G}$; the identities are the usual copycats which, like all innocent strategies, are (degenerately) view-innocent.

However, unlike for innocent strategies, the monoidal structure of $\mathbf{G}$ remains "only" monoidal in $\mathbf{V}$. This happens because, with the passage from innocent to view-innocent strategies, we lose a technical, but key, property:

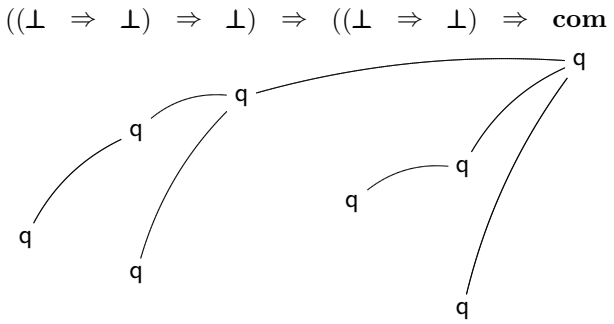
If $s \in \sigma$ for some innocent $\sigma : A \times B$, then either $\ulcorner s \urcorner \in \mathcal{L}_A$ or $\ulcorner s \urcorner \in \mathcal{L}_B$. In words, a P-*view* in $A \times B$ lives exclusively on one side or other of the product and so, since an innocent strategy is fully determined by its (P-)view function, to have an innocent strategy for $A \times B$ just means to have an innocent strategy for A and an innocent strategy for B. But a view-innocent strategy for $A \times B$ cannot generally be decomposed into a strategy for A and a strategy for B because a *view* in $A \times B$ can have moves on both sides of the product: a move on one side may depend on moves played earlier on the *other* side.

3.2 Cellular strategies

3.2.1 OP-visibility

A strategy $\sigma : A$ satisfies OP-visibility iff, for all non-empty $s \in \sigma$, the last move of s points in $\lceil s^- \rceil$. If σ is additionally view-innocent then we say that σ is ***cellular***. In one way, cellular strategies generalize innocent strategies in that they depend on the whole view, not just the P-view; but in a different way, innocent strategies generalize their cellular counterparts in that they can point in the whole P-view, not just the OP-view.

OP-visibility (and hence cellularity) can very easily be shown to be preserved by composition. However, we don't have a category since the identities are not cellular in general:



The sixth move here violates OP-visibility. Indeed, all P-moves of a copycat—except those that point to an initial move—*necessarily* violate OP-visibility.

Cellular *innocent* strategies correspond to the *transferring*, or *cellular*, λ-terms used by Padovani [9] and Loader [8] to establish decidability of observational equivalence in the minimal model and unary PCF respectively. A key property of these languages, that all (closed) terms t can be *cellularized* to an observationally equivalent (closed) cellular term $t^{©}$, allows us to build (for all types) a finite, exhaustive list of representatives of all observational equivalence classes of that type, reducing a test of observational equivalence to an exhaustive check.

Our interest in cellular strategies stems not from such decidability properties (which in any case only hold in very restricted settings) but from the simplicity of mechanizing interaction between cellular strategies. We formalize this in the next subsection.

3.2.2 The CPAM

Much as an innocent strategy is uniquely determined by its P-view function, a view-innocent strategy $\sigma : A$ is uniquely determined by its ***view function*** $[\sigma]$ which sends O-ending views to P-ending views:

$$[\sigma](s) = s' \iff \exists t \in \sigma.\ [t^-] = s \ \wedge\ [t] = s'$$

If σ is additionally OP-vis, $[t]$ can easily be calculated. Given $[t^-]$:

$[t]$ is just:

In words, the last move of t points in the OP-view of t^- and so $[t^{\leftarrow}] \sqsubseteq^{\mathsf{O}} [t^-]$. The last annotated P-arch of $[t]$ is then obtained simply by deleting the annotations under the O-arches of $[t^-]$ that lie under the pointer from the last move.

As a result, interacting *cellular* strategies need no intermediary. The output view from one can be immediately fed to the other as input with no further book-keeping required. In other words, cellular interaction can be thought of as "history free". We call this (more or less trivial) abstract machine the CPAM, the cellular PAM.

4 Cellularization

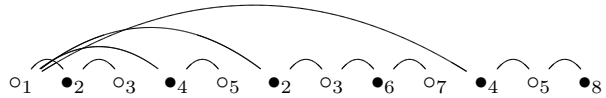
In this section, we show how to derive a cellular strategy $\sigma^{©}$ from an innocent strategy σ. This process makes the construction of output O-views completely explicit and allows us to use the CPAM to carry out innocent interaction.

4.1 Cellularizing P-views

Essentially, the cellularization procedure transforms plays so that all of Player's justification pointers point either to the immediately preceding move or to the (current) initial move. Before describing the general case, we first consider the special case of cellularizing a P-view:

- $a^{©} = a$, for initial move a
- $(s \cdot a)^{©} = s^{©} \cdot a$, if a is an O-move
- $(s \cdot b)^{©} = s^{©} \cdot b$, if b is a P-move
- finally, if we have sb where b points somewhere beyond the last move of s, let t be the OP-view $\lceil sb \rceil$ with its initial move erased; then $(sb)^{©} = s^{©}t$, where the first move of t points to the initial move of s

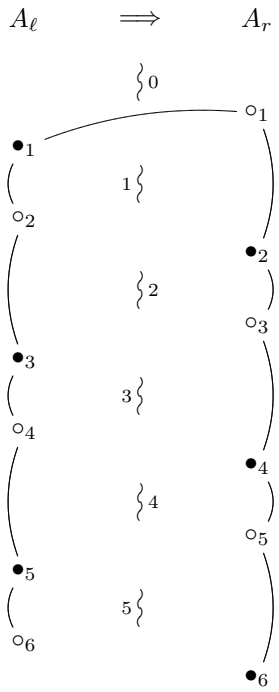
For example, the P-view $\circ_1 \bullet_2 \circ_3 \bullet_4 \circ_5 \bullet_6 \circ_7 \bullet_8$ becomes:



Note that $s^{©}$ remains a P-view. We can thus cellularize a *sequence* of P-views by $(s_1 \cdots s_i \cdots s_n)^{©} = s_1^{©} \cdots s_i^{©} \cdots s_n^{©}$, recovering another sequence of (typically much longer) P-views.

4.2 The cellular identity

As we saw earlier, the identity strategy violates OP-visiblility most of the time. We wish to define the best approximation to the identity that can "live with" the constraint of OP-visibility. To see what this means, consider a typical play s of the identity for A, which takes the following form:



We can safely assume that $s \upharpoonright_{A_\ell}$ (resp. $s \upharpoonright_{A_r}$) alternates since we only need such plays when composing id_A with another strategy. This restriction implies that $\llcorner s \lrcorner$ (resp. $\lfloor s \rfloor$) lies exclusively on the LHS (resp. RHS) when the last move of s occurs on the LHS (resp. RHS).

The output P-view of s consists of all the $\circ_i$s and $\bullet_j$s and can be seen as two interleaved, equal OP-views (of A) of opposing polarity: one on the LHS, the other on the RHS. However, the output O-view additionally inserts arches from $\overset{3}{\sim}$ and $\overset{1}{\sim}$ on the LHS; or from $\overset{4}{\sim}$, $\overset{2}{\sim}$ and $\overset{0}{\sim}$ on the RHS.

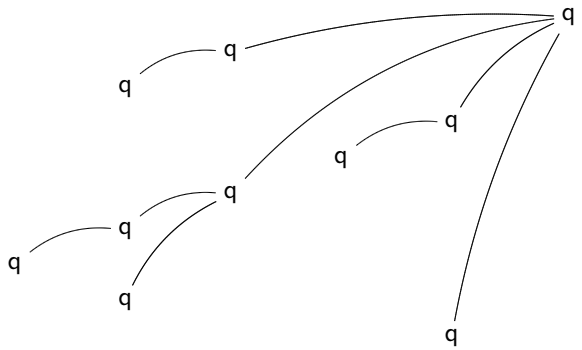
We wish to define the cellular identity in such a way that, although it cannot copycat because this would violate OP-visibility, it does nonetheless produce the same output O-views as id_A. This criterion serves as our notion of "best approximation" to the identity. To do this, instead of just copying Opponent's last move to the other side of the arrow, it must *reconstruct the whole of the output* O*-view* move-by-move. This requires the complicity of the Opponent: if, at some point during the reconstruction, Opponent plays the wrong move, the cellular identity doesn't reply.

In general, for any $s \in \mathsf{id}_A$ such that $s \restriction_{A_r}$ (or equivalently $s \restriction_{A_\ell}$) alternates, we define $s^{\copyright}$ by:

- $s^{\copyright} = t^{\copyright} a_r \llcorner s \lrcorner$, if $s = t a_r a_\ell$ ends on the LHS
- $s^{\copyright} = t^{\copyright} a_\ell \llcorner s \lrcorner$, if $s = t a_\ell a_r$ ends on the RHS

For example, the play of $\mathsf{id}_{(\bot \Rightarrow \bot) \Rightarrow \bot}$ in §3.2 becomes:

$$((\bot \Rightarrow \bot) \Rightarrow \bot) \Rightarrow ((\bot \Rightarrow \bot) \Rightarrow \mathbf{com}$$



Note that the last move here violates innocence but not view-innocence.

We now define the ***cellular identity*** for A (on $A \Rightarrow A$) by:

$$\mathsf{cid}_A = \{s^{\copyright} \mid s \in \mathsf{id}_A \ \wedge \ s \restriction_{A_r} \text{ alternates}\}.$$

4.3 Cellularization of innocent strategies

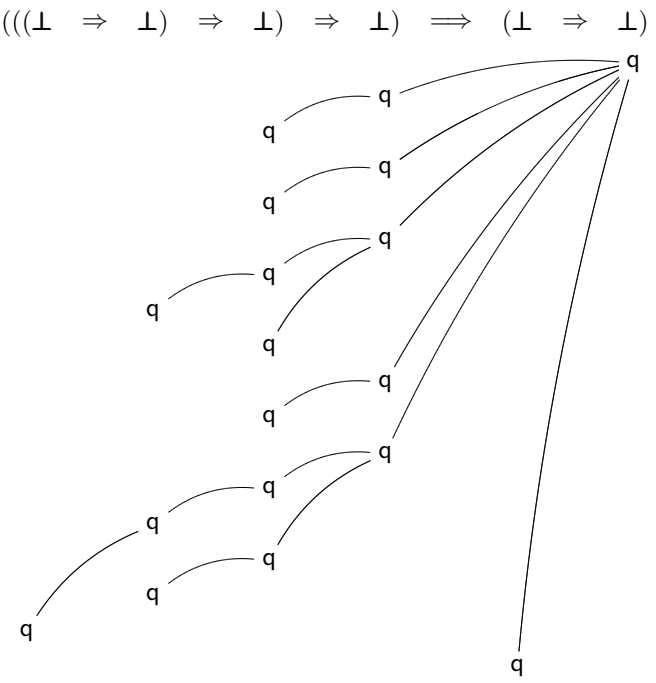
We cellularize an innocent strategy $\tau : \mathbf{1} \Rightarrow A$ (a closed term of type A) by composing with cid_A: $\tau^{\copyright} = \tau \,;\, \mathsf{cid}_A$. If $\tau : A \Rightarrow B$, we (abusively) write $\tau^{\copyright}$ for $\Lambda^{-1}(\Lambda(\tau)^{\copyright})$.

The strategy $\tau^{\copyright}$ is clearly cellular as all of its moves are played by cid_A in the defining composition. Every play $t \in \tau$ thus has a cellular counterpart $t^{\copyright} \in \tau^{\copyright}$ obtained by composing (the innocent closure of) the strategy $\{t' \in \mathcal{L}_A \mid t' \sqsubseteq t\}$ with cid_A; moreover, every $t \in \mathsf{dom}(\tau)$ has cellular counterpart $\mathsf{match}_{\mathsf{O}}(t, (t^-)^{\copyright})$. This induces an injective endofunction $(-)^{\copyright}$ satisfying $\llcorner t \lrcorner = \llcorner t^{\copyright} \lrcorner$ for all $t \in \mathcal{L}_A^{\mathsf{P}}$.

Proposition 4.3.1 *Let $\sigma : \mathbf{1} \Rightarrow A$ and $\tau : A \Rightarrow B$ (B a flat arena) be innocent strategies. Then (t) is an increasing sequence of plays in $\sigma \downarrow \tau$ if, and only if, $(t^{\copyright})$ is an increasing sequence of plays in $\sigma \downarrow \tau^{\copyright}$.*

Proof For left-to-right, we need that, for all $t_i \in (t)$, $t_i^{\copyright} \restriction_A \in \sigma \cup \mathsf{dom}(\sigma)$. But $t_i^{\copyright} \restriction_A$ is an interleaving of the same P-views of σ as $t_i \restriction_A$. So, since innocent strategies are closed under such interleavings, $t_i^{\copyright} \restriction_A \in \sigma \cup \mathsf{dom}(\sigma)$. The right-to-left direction uses the exact same reasoning. ■

An interaction between $[\![\lambda P(\mathcal{K}\mathcal{K})]\!]$, our term from §2.4, and the cellular identity can be found in table 1. This interaction produces the $t^{\copyright}$ obtained from the $t \in [\![\mathcal{K}\mathcal{K}]\!]$ that interacts with $[\![\mathcal{K}]\!]$. We thus have the following "half cellularized" interaction between $[\![\mathcal{K}]\!]$ and $[\![\mathcal{K}\mathcal{K}]\!]^{\copyright}$:



The above proposition says, and the example illustrates, that $\tau^{\copyright}$ makes the *same sequence of tests of* σ as does τ, only interspersed with repeated (and thus redundant) tests that are needed to reconstruct τ's successive output O-views. This formalizes what we meant by "complicity of the Opponent" above.

Consider now $s = t^{\copyright} \restriction_A$ for some $t^{\copyright} \in \sigma \downarrow \tau^{\copyright}$. This consists of a *sequence* of P-views of A. We apply the P-view cellularization of §4.1 to obtain a second sequence $s^{\copyright}$ of cellular P-views of A. Now, $s^{\copyright} \in \sigma^{\copyright}$ but $s^{\copyright} \notin \sigma$. Nonetheless, $s^{\copyright} \in \sigma'$ for *some* innocent $\sigma' : A$. Hence, ${}^{\copyright}t^{\copyright} = i_t s^{\copyright} \in \tau^{\copyright}$ (where i_t is the initial move of $t^{\copyright}$). So we have a (partial) injective endofunction on $\mathcal{L}_A$ mapping $t^{\copyright} \mapsto {}^{\copyright}t^{\copyright}$ such that, for all $t^{\copyright} \in \mathcal{L}_A^{\mathsf{P}}$, $\llcorner {}^{\copyright}t^{\copyright} \lrcorner = (\llcorner t^{\copyright} \lrcorner)^{\copyright}$. As an immediate consequence, we have:

Proposition 4.3.2 *$(t^{\copyright})$ is an increasing sequence of plays in $\sigma \downarrow \tau^{\copyright}$ if, and only if, $({}^{\copyright}t^{\copyright})$ is an increasing sequence of plays in $\sigma^{\copyright} \downarrow \tau^{\copyright}$.*

Putting our two propositions together, we get that (t) is an increasing sequence of plays in $\sigma \downarrow \tau$ if, and only if, $({}^{\copyright}t^{\copyright})$ is an increasing sequence of plays in $\sigma^{\copyright} \downarrow \tau^{\copyright}$. Moreover, for all the t_is of (t) such that $t_i \in \tau$, $(\llcorner t_i \lrcorner)^{\copyright} = \llcorner {}^{\copyright}t^{\copyright} \lrcorner$.

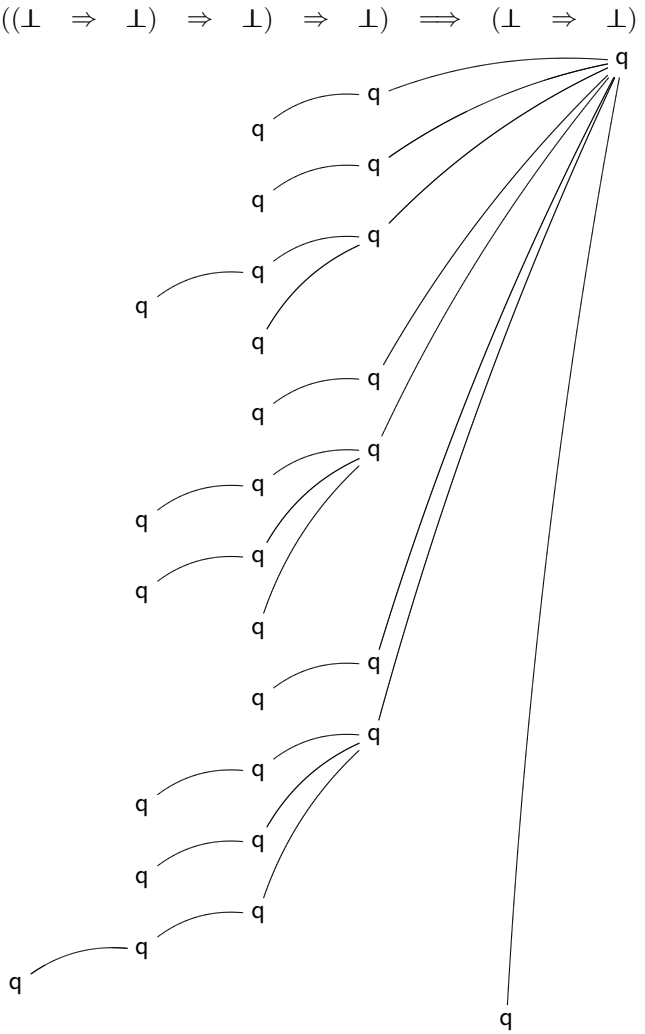
Table 1: An interaction of $[\![\mathcal{K}\mathcal{K}]\!]$ and the cellular identity.

In words, for τ's successive output O-views t in $\sigma \downarrow \tau$, we have corresponding output O-views ${}^{©}t^{©}$ of $\tau^{©}$ in $\sigma^{©} \downarrow \tau^{©}$ (and vice versa). So, σ's successive output P-views s in $\sigma \mid \tau$ become $\sigma^{©}$'s output P-views $s^{©}$ in $\sigma^{©} \downarrow \tau^{©}$ (and vice versa). In summary:

Theorem 4.3.3 *The CPAM interaction between $\sigma^{©}$ and $\tau^{©}$ simulates the innocent interaction between σ and τ.*

So, at the (considerable) cost of cellularizing, we can indeed compute innocent interactions without any kind of referee.

To complete our running example, we give below the fully cellularized interaction between $\mathcal{KK}^{©}$ and $\mathcal{K}^{©}$.



5 Conclusions and future work

The above theorems tell us that the CPAM can be used to implement innocent interaction. The CPAM has certain advantages—simplicity and no referee—but pays the price of very long interactions. More significantly, the cellularization process explains the extraordinarily complex combinatorics of innocent interaction by emphasizing that each *move* of an innocent interaction really corresponds to a P-*view for the other strategy*. We thus gain a higher-level perspective which may lead to more elegant (and efficient!) presentations of game semantics.

Interesting questions also revolve around the notion of view-innocence: as yet, we have no good syntax for such strategies, not even a notion of Böhm tree. Such a syntax could lead to a better understanding of Algol-like languages (that satisfy the visibility condition but not innocence) since the most significant such strategies *are* view-innocent. So, a language for view-innocent strategies could lead to a new syntactic perspective on languages with base type references.

References

[1] T. Coquand. A semantics of evidence for classical arithmetic. *Journal of Symbolic Logic*, 60(1):325–337, 1995.

[2] P.-L. Curien and H. Herbelin. Computing with abstract Böhm trees. In *Fuji International Symposium on Functional and Logic Programming*. World Scientific, 1998.

[3] P.-L. Curien and H. Herbelin. Abstract machines for dialogue games. *Panoramas et synthèses*, 2006. To appear.

[4] V. Danos and R. Harmer. The anatomy of innocence. In *Proceedings, 10th Annual Conference of the European Association for Computer Science Logic*. Springer Verlag, 2001.

[5] V. Danos, H. Herbelin, and L. Regnier. Game semantics & abstract machines. In *Proceedings, 11th Annual IEEE Symposium on Logic in Computer Science*. IEEE Computer Society Press, 1996.

[6] J. M. E. Hyland and C.-H. L. Ong. On full abstraction for PCF: I, II and III. *Information and Computation*, 163:285–408, 2000.

[7] O. Laurent. Polarized games. *Annals of Pure and Applied Logic*, 130(1–3):79–123, December 2004.

[8] R. Loader. Unary PCF is decidable. *Theoretical Computer Science*, 206:317–329, 1998.

[9] V. Padovani. Decidability of all minimal models. In *TYPES '95: Selected papers from the International Workshop on Types for Proofs and Programs*, London, UK, 1996. Springer-Verlag.